به نام خدا

آموزش نرم افزار CLIPS

به زبان ساده – قسمت سوم

یادآوری

در قسمت‌های قبلی با تعریف ***(assert)*** ، مشاهده ***(facts)*** ، حذف ***(retract)*** حقایق و تعریف ***(defrule)*** ، مشاهده ***(rules)*** و اجرای ***(run)*** قوانین آشنا شدیم و همچنین نحوه تعریف حقایق اولیه ***(deffacts)*** فراخوانی آنها ***(reset)*** ، استفاده از نویسه های جایگزین در الگو تعریف ***(?)*** ، نحوه استفاده از متغیر ها ***(?var)*** و عملگرهای منطقی ***(and , or , not)*** و ریاضی ***(* , / , - , +)*** را نیز فرا گرفتید..

مثال: عبارت ریاضی زیر

22 + 5 * 41 - 10 / 4

در CLIPS به شکل زیر تعریف می شود:

```
CLIPS> (+ 22 (- (* 5 41) (/ 10 4)))
224.5
```

ارتباط با کاربر

گاهی اوقات برای اینکه یک سیستم خبره کارایی بهتری داشته باشد، باید بتواند با کاربر تعامل داشته باشد و اطلاعاتی را از کاربر دریافت کند.

- دستور ***(read)*** : از این دستور برای دریافت اطلاعات از کاربر استفاده می شود.

مثال: یک قاعده که نام کاربر را گرفته و بعنوان یک داده ذخیره کند

```
CLIPS> (defrule Get_name
=>
(printout t "What's your name? " crlf
"My name is: ")
(assert (user_name (read))))
```

پس از اجرای قاعده بالا *(run)* ، هیچ شرطی چک نشده و پیام "نام شما چیست؟" بنمایش در میاد. از انجا که از دستور *(read)* استفاده شده، سیستم منتظر ورود داده توسط کاربر میشود و پس از فشردن اینتر حقیقت جدید به لیست حقایق اضافه خواهد شد.

متغیرهای چند مقداری: زمانی که از یک 'نویسه جایگزین' (؟) و یا یک متغیر در یک الگو استفاده می‌کنیم، آن را تنها با یک داده منطبق می‌شود. اما اگر از علامت $ قبل از متغیرها استفاده کنیم، می توانند چندین مقدار را دربر بگیرد

$? [نام متغیر]

- تابع ***bind*** : تاکنون برای مقداردهی یک متغیر آن را در الگو می نوشتیم، اما با استفاده تابع ***bind*** نیز میتوان متغیرها را مقداردهی نمود.

(bind نام متغیر مقدار **)**

مثال: فرض کنید عدد پی را به عنوان یک حقیقت داریم.

```
CLIPS> (assert (circle pi 3.14))
```

و می‌خواهیم قاعده ای بنویسیم که با گرفتن عدد پی از لیست حقایق و شعاع یک دایره از کاربر، مساحت یک دایره را حساب کند، نمایش دهد و در نهایت حقیقت مساحت دایره را اضافه کند.

```
CLIPS> (defrule circular_area
(circle pi ?p)
=>
(printout t "enter radius :")
(bind ?r (read))
(bind ?area (* ?p (* ?r ?r)))
(printout t  crlf "circle area : " ?area crlf)
(assert (circle area ?area)))
```

پس از اجرای این قاعده، متغیر ?p مقداردهی شده (با ۳.۱۴) و سپس پیام "شعاع را وارد کن" به کاربر نمایش داده می‌شود. پس از ورود شعاع توسط کاربر، مساحت دایره (شعاع * شعاع * ۳.۱۴) محاسبه و در متغیر ?area قرار میگیرد. در خط بعدی پیام مساحت دایره با مقدار بدست امده نمایش و در خط آخر حقیقت مساحت دایره به لیست حقایق اضافه می‌شود.

- دستور ***defglobal*** : متغیرهای سراسری را در این ساختار تعریف میکنیم، بطوری که نام متغیر سراسری بین دو علامت * قرار می گیرد.

(defglobal ?*نام متغیر* **)**

مثال:

```
CLIPS> (defglobal
?*var1* = 22
)
```

تعریف حقایق بر اساس ساختار

در هنگام تعریف حقایق میتوانید آنها را از قبل با یک الگو ی یا ساختار خاص مشخص کرده باشید که مثلا چه فیلدهایی باید در این حقیقت باشد، فیلدها از چه نوع باشند، تک مقداری یا چند مقداری باشند و ...

- دستور ***deftemplate*** : تعریف الگو (ساختار) برای حقایق، توسط این دستور صورت می‌گیرد.

(deftemplate نام الگو "توضیحات"
[نوع داده ای type] نام فیلد Multislot یا Slot
)

نکته: slot نشان دهنده تک مقداری و multislot نشان دهنده چند مقداری بودن (صفر و بیشتر) فیلد است. نوع داده ای نیز می تواند (integer, string , float,...) باشد. البته نوع داده ای اختیاری است.

مثال: تعریف الگو برای حقیقت ماشین

```
(deftemplate  car  “Attributes of one car”
(multislot name)
(slot color)
(slot model)
(slot company))
```

در حال حاظر شما یک ساختار یا الگو برای حقیقت ماشین ساخته اید. هر زمان که بخواهید حقیقت ماشین را تعریف کنید (توسط دستور assert) شما می‌بایست بر اساس این ساختار فیلدها را مشخص کنید.

اگر فیلدی را در این ساختار تعریف نکنید با مقدار تهی یا nil مشخص خواهد شد

```
CLIPS> (assert (car))
==> f-1    (car (name) (color nil) (model nil) (company nil))
```

و

```
CLIPS> (assert ( car
 (name samandLX)
(color black)(model 2014)(company IR)))
==> f-2    (car (name samandLX) (color black) (model 2014) (company IR))
```

IF-Else

یک سیستم خبره باید بتواند بر اساس شرایط، عملیات خاصی را انجام بدهد، از این رو از دستورات شرطی استفاده می‌کنیم که ساختاری مشابه سایر زبان ها دارد:

(if شرط
then عملیات
[else عملیات **])**

مثال: فرض حقیقت سن از پیش تعریف شده

```
CLIPS> (assert (age 23))
```

می‌خواهیم اگر سن کمتر از ۲۵ بود پیام "بله" و در غیز این صورت پیام "خیر" نمایش داده شود

```
CLIPS> (defrule age-value
(age ?age)
=>
(if (< ?age 25)
then
(printout t "yes" crlf)
else
(printout t "no" crlf)))
```

پس از اجرا

```
CLIPS> (run)
yes
```

کمی حرفه‌ای تر

همان طور که می دانید CLIPS تنها جهت پیاده سازی سیستم خبره است. سیستم خبره‌ای که شما آن را در ذهن و یا روی کاغذ طراحی کرده اید و این طراحی بر اساس تحقیق و پژوهشی است که شما در یک حوزه و از فرد خبره داشته‌اید.

بنابراین جهت پیاده سازی یک سیستم خبره ابتدا شما می بایست به پژوهش و گفتگو با فرد خبره در آن حوزه بپردازید و کسب دانش کنید. سپس شما باید سناریو سیستم خود را بنویسید. این سناریو می تواند همان درخت تصمیم گیری باشد. طبق سناریو از پیش تعیین شده شما می‌توانید داده هایی را از کاربر بگیرید و بر اساس آن داده ها، خبرگی سیستم را که قوانین آن را کنترل می‌کنند به نمایش در آورید.

آنچه تا کنون خواندید همگی مثال‌های ساده جهت یادگیری نرم‌افزار CLIPS بود. اما اگر می‌خواهید یک سیستم خبره را بسازید باید مراحل گفته شده را دنبال کنید.

« ما در تمرین شماره یک (قسمت چهارم)، مثالی از طراحی، ایجاد درخت تصمیم، سناریو نویسی و پیاده سازی در نرم افزار CLIPS یک سیستم خبره ساده را بیان کرده ایم »

پایان قسمت سوم